




Doc Code: AP.PRE.REQ

PTO/SB/33 (07-05)

Approved for use through xx/xx/200x. OMB 0651-00xx

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

PRE-APPEAL BRIEF REQUEST FOR REVIEW		Docket Number (Optional) 0081004.00176US1 (RSA-054)	
Doc Code: AP.PRE.REQ	Application Number 10/010,769-Conf. #3025	Filed December 4, 2001	
	First Named Inventor John BRAINARD et al.		
	Art Unit 2131	Examiner L. Chai	
<p>Applicant requests review of the final rejection in the above-identified application. No amendments are being filed with this request.</p> <p>This request is being filed with a notice of appeal.</p> <p>The review is requested for the reason(s) stated on the attached sheet(s). Note: No more than five (5) pages may be provided.</p> <p>I am the</p> <p><input type="checkbox"/> applicant /inventor.</p> <p><input type="checkbox"/> assignee of record of the entire interest. See 37 CFR 3.71. Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96)</p> <p><input type="checkbox"/> attorney or agent of record. Registration number _____</p> <p><input checked="" type="checkbox"/> attorney or agent acting under 37 CFR 1.34. Registration number if acting under 37 CFR 1.34. <u>32,590</u></p> <p> Signature Eric L. Prah Typed or printed name (617) 526-6000 Telephone number October 12, 2006 Date</p> <p>NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below*.</p> <p><input type="checkbox"/> *Total of <u>1</u> forms are submitted.</p>			

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being deposited with the U.S. Postal Service on the date shown below with sufficient postage as First Class Mail, in an envelope addressed to: MS AF, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Dated: October 12, 2006

Signature:  (Maureen DiVito)



The Examiner rejected claims 1-28 and 30-31 as being unpatentable over Weiss (U.S. Patent 4,885,778) in view of Kocher (U.S. 6,539,092). The Examiner admits that Weiss does not disclose first and second generation values, which are values indicative of a previous number of authentication code generations. The Examiner argues that Kocher supplies these elements, in the form of transaction counter C. However, even if one assumes that transaction counter C is equivalent to first and second generation values, neither Kocher nor Weiss teach or suggest using that value in the way required by claim 1. Specifically, neither Kocher nor Weiss teach or suggest:

...generating an authentication code by combining the stored secret, the dynamic value, the first generation value, and the PIN...

To explain why neither reference teaches or suggests “combining” Kocher’s transaction counter C with Weiss’s stored secret, dynamic value, and PIN, we first address what is meant by “combining.” The specification and the claims both use the word “combining” in its normal sense, i.e., meaning to merge, or to bring into such close relationship as to obscure individual characters. For example, the specification provides some non-limiting examples of “combining” the secret, dynamic value, and generation value that are consistent with the word’s plain meaning:

The combination of the secret (K) the dynamic value (T) and the generation value (N) may take place in any order and may use one or more various combination methods. For example, in one simplistic embodiment, the values (K,T,N) are EXCLUSIVE-ORED with each other to arrive at a resulting authentication code. In another embodiment, the values (K,T,N) are provided as input to a one-way function... In one of these embodiments, the combination function 130 is designed such that each different generation value (N) that is combined with a constant stored secret (K) and a dynamic value (T) results in a different authentication code value ([0039]).

In these examples, the stored secret, dynamic value, and generation value are merged to produce a single value: the authentication code.

In contrast, Kocher does not teach or suggest generating an authentication code by combining transaction counter C with any other value. Rather, Kocher discloses using transaction counter C to update a secret key K_C , e.g., an “authentication code,” but Kocher does not perform the update process by combining C with any other values. Instead, Kocher’s update process involves using a rule based on the value of C to select one of “two forward cryptographic transformations (F_A and F_B) and their inverses (F_A^{-1} and F_B^{-1})” (col. 2, lines 60-61), applying the selected transformation to secret K_C , and incrementing C. Kocher describes the process of updating secret key K_C relative to Fig. 1:

At step 100, the client is initialized or personalized with a starting counter $C=0$ and a starting state having a starting secret value $K_C=K_0$. At step 110, the device performs the first transaction, using K_C (or a key derived from K_C)...

After step 110, the client device's secret value K_C is updated by applying the function F_A and the counter C is incremented, i.e. by performing $C \leftarrow C+1$ and $K_C \leftarrow F_A(K_C)$. (Thus at step 111, $C=1$ and $K_C=F_A(K_0)$.) The updated value of K_C is used to perform a transaction at step 111 (col. 4, lines 37-41 and lines 52-56).

So, before the transaction at step 110, $C=0$ and $K_C=K_0$. After the transaction at step 110, Kocher increments C , so that $C=1$, and applies F_A to starting secret K_0 thereby generating an updated secret K_C . Then K_C is used for a new transaction at step 111. Kocher describes updating secret K_C again after the new transaction:

After step 111, C is incremented again and F_A is again applied to K_C , i.e. by performing $C \leftarrow C+1$ and $K_{C=2} \leftarrow F_A(K_C)$, yielding the secret key used at step 112 (col. 4, lines 56-59).

In other words, after the transaction at step 111, Kocher increments C , so that $C=2$, and applies F_A to the previously used secret K_C thereby generating a newly updated secret K_C . Kocher goes on to describe further updating K_C by applying additional functions to it after additional transactions:

The same pair of operations ($C \leftarrow C+1$ and $K_C \leftarrow F_A(K_C)$) are similarly applied between steps 112 and 113, and between steps 113 and 114.

...After the transaction at step 115, K_C is updated using function F_B by incrementing C and computing $K_{C=6} \leftarrow F_B(K_C)$. After the transaction at step 116, the secret value for transaction 117 is computed by applying the function F_B^{-1} to K_C (col. 4, lines 59-61 and col. 6, lines 1-5).

So, Kocher updates secret key K_C by applying one of the cryptographic functions F_A , F_B , F_A^{-1} , or F_B^{-1} to secret key K_C .

As mentioned above, Kocher selects which cryptographic function to apply to K_C using a rule based on the value of C : "The choice of which function to apply in any particular state transition can be determined solely as a function of C " (col. 5, lines 33-35). To determine which function to apply, Kocher first initializes a temporary counter variable V to the value of C , and then determines which function to apply to K_C using a rule based on the value of V .

At step 230, the device tests whether the variable V is equal to the quantity $2N-3$. If equal, function F_{A-1} should be applied, and processing proceeds to step 235 where the device increments C and updates K_C by computer $K_C \leftarrow F_{A-1}(K_C)$. Otherwise, at step 240, the device tests whether the variable V is equal to the quantity $2(2N01)$. If equal, function F_{B-1} should be applied...(col. 6, lines 8-14).

Kocher goes on to describe the rest of the rules, based on the value of V (i.e., C), for determining which one of the functions F_A , F_B , F_A^{-1} , or F_B^{-1} to apply to secret K_C (col. 6, lines 14-24). As Fig. 2 illustrates, each time Kocher applies a function to secret K_C , he also increments C .

So, Kocher uses rules based on the value of transaction counter C, and increments C. Neither of these actions constitute combining C with another value e.g., an authentication code.

The Examiner argues otherwise, but his arguments are flawed. In the Final Office Action dated April 12, 2006, the Examiner argues:

Kocher thoroughly teaches a security key update process to securely computing, for example, a message authentication code by combining, at least a transaction counter and a secret value (Kocher: Column 4 Line 39-44) (Final Office Action, p. 3).

However, the section of Kocher to which the Examiner refers does not describe generating secret key K_C by combining transaction counter C with a “secret value.” Instead, the cited section states:

At step 110, the device performs the first transaction, using K_C (or a key derived from K_C). The key can be used in virtually any symmetric cryptographic transaction. (For example, such a transaction could involve, without limitation, computing or verifying a MAC (Message Authentication Code) on a message...) (col. 4, lines 39-44).

So, the section describes an intended use of secret key K_C , e.g., for cryptographic transactions. Despite the Examiner’s assertions, the section does not disclose “securely computing” an authentication code by “combining, at least a transaction counter and a secret value,” and indeed provides absolutely no insight as to how Kocher generates K_C . Based on the cited section, it is even not clear to what “secret value” the Examiner refers.

In the Advisory Action dated July 11, 2006, the Examiner argues that “Kocher is relied upon to provide combining another data, i.e., a first generation value (Kocher: Column 2 Line 47-53, Column 5 Line 4-5 and Column 3 Line 54-60: a transaction counter is considered as equivalent to a first generation value”). Again, however, the cited sections do not describe “combining” transaction counter C with any other value. The first cited section describes using transaction counter C on the server side of a transaction to re-derive the client’s secret key:

To perform a transaction with the client, the server obtains the client’s current transaction counter (or another key index value). The server then performs a series of operations to determine the sequence of transformations needed to re-derive the correct session key from the client’s initial secret value. These transformations are then performed, and the result is used as a transaction session key (or used to derive a session key) (col. 2, lines 47-53).

The “sequence of transformations” to which Kocher refers is the application of the functions F_A , F_B , F_A^{-1} , and F_B^{-1} to K_C . The second section the Examiner cites provides an example of computing K_C for a new transaction:

After the transaction at step 116, the secret value for transaction 117 is computed by applying the function F_B^{-1} to K_C (col. 5, lines 4-5).

This is simply a specific application of one of the above-described rules for selecting a function based on the value of C. Applying the function F_B^{-1} does not in any way entail combining C with another value. The third section the Examiner refers to describes another variable, D, that Kocher uses during his key update process:

The client also has a (typically non-secret) index or transaction counter C, which may be initialized to zero. An additional parameter is an index depth D. The value of D may also be non-secret, and (for example) may be client-specific or may be a system-wide global constant. The value of D determines the cycle length of the key update process (col. 3, lines 54-60).

The index depth D simply determines “the number of transactions that can be performed before the end of the process occurs” (col. 5, lines 50-51). Thus the third cited section simply describes two variables used in the key update process.

In summary, even if one were to combine the teachings of Kocher with the teachings of Weiss, the result would not be the claimed invention which requires “generating an authentication code by combining the stored secret, the dynamic value, the first generation value, and the PIN.” Moreover, the Examiner has pointed to nothing, nor could we anything in Weiss or Kocher, that suggests or would lead one skilled in the art to employ the transaction counter in the way that is required by the claims (i.e., combining it with a stored secret, a dynamic value, and a PIN).

Similar reasoning applies to claim 17 which recites:

a combination subsystem generating an authentication code by retrieving the secret from the memory element and combining the secret and the dynamic value from the dynamic value subsystem, the PIN received by the PIN subsystem, and the generation value from the generation value subsystem.

As discussed above, Kocher does not teach or suggest generating an authentication code by combining a generation value with any other value. Thus, Kocher also does not teach or suggest a combination subsystem that performs this function.

For the reasons stated above, we submit that the application is in condition for allowance, and therefore ask that the claims be allowed to issue.